POWER FLOW SOLUTION FOR RADIAL DISTRIBUTION NETWORKS

CHUNG TZE LING

ELECTRICAL & ELECTRONICS ENGINEERING
UNIVERSITI TEKNOLOGI PETRONAS

JUNE 2009

Power Flow Solution for Radial Distribution Networks

by

Chung Tze Ling

Final Report submitted in partial fulfilment of
the requirements for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

JUNE 2009

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Power Flow Solution For Radial Distribution Networks

by

Chung Tze Ling

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(ELECTRICAL & ELECTRONICS ENGINEERING)

Approved by,

(Dr. Ramiah Jegatheesan)
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

June 2009

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

____
CHUNG TZE LING

ABSTRACT

Power flow analysis on distribution systems is not having enough attention as compared to transmission systems. Generally, distribution networks are radial and the resistance to reactance ratio, R/X is high. By making use of the special structure of radial network, a simple method is first developed to obtain the connection matrix and hence the nodes beyond any specified branch, which is the branch-node matrix by using the special structure of radial distribution network. The necessary formulas are derived to calculate the receiving-end voltage in terms of the sending-end voltage and the receiving-end line flows. At each and every of the iterations, receiving-end voltages are updated by computing the line losses. The main aim of this project is to attain a simple power flow method which is suitable for solving radial distribution networks and develop the necessary MATLAB programme. The specially designed MATLAB programme to solve radial distribution networks was successfully developed and tested on several standard radial distribution networks. Meanwhile, Newton-Raphson power flow method is developed to test the radial networks for strengthening the validity of the results obtained. Besides, comparisons are made between both Newton-Raphson power flow method and proposed approach. The proposed approach for radial distribution network can be implemented on any practical data.

ACKNOWLEDGEMENT

First of foremost, I would like to express my greatest gratitude to my Final Year Project Supervisor, Prof. Dr. Ramiah Jegatheesan for his advices, comments, ideas, valuable time and his support and endless guidance towards me for completing my project. Besides, I would like to thank my coursemate, Tan Mao Cheng for sharing with me on his MATLAB knowledge and also his time to guide me on troubleshooting MATLAB programming for this project. In addition, I would like to express my gratitude for MATLAB programme given by University authorities. Special thanks to my beloved parents for their endless support.

TABLE OF CONTENTS

CERTIFICATION OF APPROVAL i

CERTIFICATION OF ORIGINALITY ii

ABSTRACT iii

ACKNOWLEDGEMENT iv

LIST OF FIGURES vii

LIST OF TABLES viii

LIST OF ABBREVATIONS ix

CHAPTER 1: INTRODUCTION 1

1.1 Background of Study 1

1.1 Problem Statement 2

1.2 Objective 2

1.3 Scope of Study 2

CHAPTER 2: LITERATURE REVIEW 3

2.1 Power Distribution 3

2.2 Power Flow Analysis 3

2.3 MATLAB Analysis 4

2.4 Newton-Raphson Power Flow Solution . 4

2.5 Power Flow Analysis Software . . . 5

2.6 Power Flow Methods for Radial Distribution Networks . . . 5

CHAPTER 3:	**METHODOLOGY**		8
	3.1	Research Methodology . . .	8
	3.2	Tool	9
	3.3	Newton-Raphson Power Flow Methodology .	9
	3.4	Assumption	12
	3.5	Proposed Solution Approach . . .	13
		3.5.1 Connection Matrix, C . . .	13
		3.5.2 Branch-node Matrix, BN . .	15
		3.5.3 Proposed Solution Algorithm for Radial Distribution Networks . .	17
CHAPTER 4:	**RESULTS AND DISCUSSION** . . .		19
	4.1	Results	19
		4.1.1 Voltage Solution . . .	19
	4.2	Discussion	21
		4.2.1 Load Flow Solution Analysis . .	21
		4.2.2 Comparison of Proposed Approach and Newton-Raphson Power Flow Method	24
		4.2.3 Proposed Approach . . .	24
		4.2.4 Accommodating Voltage Dependent Loads	25
CHAPTER 5:	**CONCLUSION AND RECOMMENDATION.** .		27
	5.1	Conclusion	27
	5.2	Recommendation	27
REFERENCES	. . . .		28
APPENDICES	. . . .		29

LIST OF FIGURES

Figure 1: Project Flow Chart 8

Figure 2: Newton-Raphson Power Flow Method Solution Flow Chart 10

Figure 3: Single-line diagram of a 12-bus radial distribution network 12

Figure 4: Single-line diagram of a 15-bus radial distribution network 13

Figure 5: Connection Matrix, **C** for 15-bus system 14

Figure 6: Branch-Node Matrix, **BN** for 15-bus system 16

Figure 7: Flow Chart for Radial Distribution Networks 17

Figure 8: Single line diagram of 28-bus radial distribution network 23

Figure 9: Flow Chart for Identification of Nodes and Branches beyond a Particular Node [3] 46

Figure 10: Flow Chart for Identifying Nodes Beyond All The Branches [4] 47

LIST OF TABLES

Table 1: Branch number, sending-end and receiving-end nodes 14

Table 2: Load flow solution of 15-bus 19

Table 3: Load flow solution of 28-bus 20

Table 4: Load flow solution of 30-bus 20

Table 5: Comparison of Newton-Raphson power flow method with proposed approach by number of iteration and computational time 21

Table 6: Load flow solution of 28-Bus from [4] 22

Table 7: Common Values for the Exponents for Different Static Load Models 26

Table 8: Final Year Project – Semester I 31

Table 9: Final Year Project – Semester II 32

Table 10: Data for 15-bus Radial Distribution Network [3] 43

Table 11: Data for 28-bus Radial Distribution Network [4] 44

Table 12: Data for 30-bus Radial Distribution Network [17] 45

LIST OF ABBREVATIONS

NB	total number of buses/nodes
jj	branch number, j = 1,2,3,...,LN1
$I_L(i)$	load current at bus i, i = 1,2,3,...,NB
Z(jj)	impedance of branch jj
I(jj)	current of branch jj
$P_L(i)$	real power load at *i*-th node
$Q_L(i)$	reactive power load at i-th node
IS(jj)	sending-end node of branch jj
IR(jj)	receiving-end node of branch jj
C	connection matrix
BN	branch-node matrix

CHAPTER 1
INTRODUCTION

1.1 Background of Study

Power flow analysis of distribution systems has been given a very little attention as compared to power flow analysis of transmission systems. Distribution networks are typically of two types, radial or interconnected. A radial network leaves the station and passes through the network area with no normal connection to any other supply. This is typical of long rural lines with isolated load areas. An interconnected network is generally found in more urban areas and will have multiple connections to other points of supply. Distribution systems have special features such as radial structure and high resistance to reactance (R/X) ratios. Hence, the distribution networks are ill-conditioned in nature. Generally, the 11kV rural distribution feeders are radial and too long. The voltages at the far end of the feeders are very low with very high voltage regulation. Moreover, the losses of transmission and distribution are as high as 20% to 40% of the total power generation. The power flow solution method has to be high speed and low storage for large systems. Besides, they should be of high reliability, flexible, simple and able to reach convergence.

1.2 Problem Statement

Power flow solution of radial distribution networks is getting more important. Due to the high R/X ratio of distribution networks, conventional power flow solution methods may result in ill-conditioned Jacobian matrix. The selection of a power flow solution method for a practical radial distribution network system is often difficult. Therefore, various power flow methods need to be studied to arrive at the suitable and efficiently convergent power flow method to solve radial distribution networks.

1.2 Objective

The aim of this project is to acquire a suitable and efficient power flow method to solve radial distribution networks and develop the necessary MATLAB programme.

1.3 Scope of Study

- Studies of various power flow methods
- Arriving at suitable power flow method for radial distribution networks
- Development of suitable power flow method in MATLAB programming
- Carry out case studies for radial distribution networks

CHAPTER 2
LITERATURE REVIEW

2.1 Power Distribution

The distribution system connects the distribution substations to the consumers' service-entrance equipment. Distribution systems are both overhead and underground. The range for primary distribution lines are usually 4 to 34.5 kV and supply the load in a well-defined geographical area. There are some small industrial customers are served by the primary feeders directly. The secondary distribution network minimizes the voltage for utilization by commercial and residential consumers. [1]

2.2 Power Flow Analysis

Power flow analysis is the most essential study carried out by the power companies. Power flow studies, frequently referred to as load flow, are the backbone of power system analysis and design. It is necessary for planning, operation, economic scheduling and exchange of power between utilities. In addition, power flow analysis is required for many other analyses such as transient stability and contingency studies. In power system, powers are known rather than currents. Thus, the resulting equations in terms of power, known as the power flow equation, become non-linear and must be solved by iterative techniques. [1]

The problem consists of determining the magnitudes and phase angle of voltages at each bus and active and reactive power flow in each line. The system is assumed to be operating under balanced conditions and a single-phase model is used in solving a power flow problem. There are four quantities are linked with each bus which are voltage magnitude $|V|$, phase angle δ, real power P and reactive power Q. There are three types of system buses which are slack bus, load buses and regulated buses. In any system, one of the generator buses will be the slack bus. It is usually taken as reference where the magnitude and phase angle of the voltage are specified. [1]

For load buses, which are also known as P-Q bus, the active and reactive powers are specified. The magnitude and the phase angle of the bus voltages are unknown. Regulated buses, which are also known as P-V buses are the generator buses. They are voltage-controlled buses. The real power and voltage magnitude are specified at these buses. However, the phase angles of the voltages and the reactive power need to be determined. Usually, the limits on the value of the reactive power are also specified. [1]

2.3 MATLAB Analysis

A practical power system must be safe, reliable and economical. Therefore, various analyses must be performed to design and operate an electrical system. Before the system analysis, all components of electrical power system need to be modeled. The parameters of a radial distribution network can be calculated after reviewing the concepts of power. The MATLAB environment allows a direct transition from mathematical expression to simulation which includes evaluation of transmission line parameters and power flow analysis. [1]

2.4 Newton-Raphson Power Flow Solution

Newton-Raphson is one of the efficient power flow conventional method. It is found to be more efficient and practical for large power systems. This method is mathematically superior to another conventional power flow solution method, Gauss-Siedel because of its quadratic convergence.

The number of iterations required to acquire a solution is independent of the system size. However, each of the iteration requires more functional evaluations. The power flow equation is formulated in polar form since in the power flow problem real power and voltage magnitude are specified for the voltage-controlled buses.

Decoupled Newton-Raphson method uses two sub-matrices of Jacobian matrix, is significant improvement over Newton-Raphson method. The coefficient matrices are constant in Fast Decoupled Power Flow (FDPF) method. Since factorization is done only once, FDPF method is well suited for large scale power systems. FDPF method linked more number of iterations and slower convergence characteristic.

2.5 Power Flow Analysis Software

The power flow analysis software which is commonly used in industries is such as EDSA and ERACS softwares. The EDSA software is used because of its precision and reliability while ERACS is user friendly and it has powerful graphical user interface. [2] Networks for simulation by ERACS may be either radial or fully interconnected systems or both. ERACS can be used in power systems engineering for modeling, load flow analysis, fault analysis, system stability and protection co-ordination on all.

2.6 Power Flow Methods for Radial Distribution Networks

Power flow analysis distribution received very less attention as compared to power flow analysis of transmission systems. Radial structure with high R/X ratio cause distribution networks to be poor conditioned which cause the conventional power flow methods such as Newton-Raphson and Fast Decoupled Load Flow (FLDF) methods to be inefficient at solving such networks. [3,4]

In recent times, researchers have focus on acquiring the solution of distribution networks. The modified versions of the conventional power flow methods to solve ill-conditioned power networks with high R/X ratios have been suggested by many researchers. [3]

Most of the distribution line has short length, which is less than 80km in general. Besides, the voltage level is at the transmission systems is higher than the distribution systems. Hence, the π model is not used because the line charging shunt capacitance is of negligible order. Therefore, the short line model is used to represent the line sections for the distribution systems. Here, the distribution line sections are modeled as series of impedances, $Zi = Ri + jXi$. [18]

Assorted power flow methods of obtaining power flow solution for radial distribution networks are reported in literature [3-17]. Kersting [5,6] suggested to use ladder-network theory to solve radial distribution networks, which work very well. However, according to Stevens et. al. [7], the ladder technique is inefficient as it did not converge in five out of twelve cases studied although it is found to be the fastest.

Shirmohammadi et. al. [8] introduced a method based on the direct application of Kirchhoff's voltage and current laws. The numerical performance of the solution method is improved by developing branch-numbering scheme by them.

Baran and Wu [10] obtained power flow solution in distribution system by the iterative solution of three fundamental equations which are real power, reactive power and voltage magnitude. Chain rule is applied to compute the system Jacobian matrix. There is only evaluation of simple algebraic expressions and no trigonometric functions are involved in the Jacobian matrix and also the mismatches.

The very fast decoupled distribution power flow that is proposed in [11] is appealing as it does not require any Jacobian matrix. The method in [12] has proposed a method that gives a solution for bus voltage magnitude only. In [13], a direct solution was proposed for solving radial distribution networks but the main limitation of their method is that no node in the network in the junction having more than three branches.

A new power flow method for solving the radial distribution networks by deriving the fundamental equations by using a single-line equivalent is proposed by Jasmon and Lee [14,15]. Das et. al. [3] introduced a load flow technique by calculating the total real and reactive power fed through any node to solve radial distribution networks.

They have introduced a unique node, branch and lateral numbering scheme which assist to calculate exact real and reactive power flows fed through any node and receiving-end voltages [4]. Their method always guarantees convergence of any type of practical radial distribution network with a practical R/X ratio. Another advantage of their method is that it involves less computer memory and easily manages the breaking up of the composite loads [3].

A simple and efficient method was introduced by Ulas Eminoglu and M.Hakan Hocaoglu [17], which is based on the forward and backward voltage updating by using polynomial voltage equation for each branch and backward ladder equation. The ability to converge and reliability of the method is being compared with Ratio-Flow method, which is based on the classical forward-backward ladder method. It is being compared for different loading conditions, R/X ratios and different source voltage levels, under the wide range of loads exponents.

CHAPTER 3
METHODOLOGY

3.1 Research Methodology

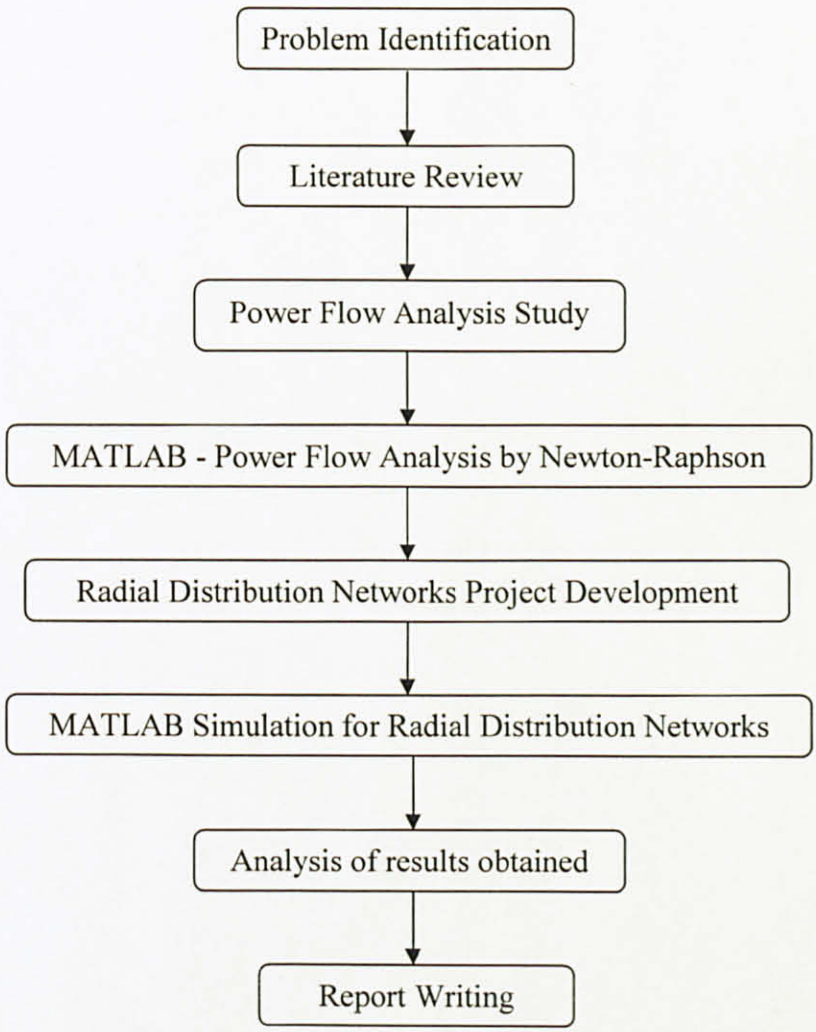

Figure 1: Project Flow Chart

3.2 Tool

The software tool used is MATLAB R2008a (Version 7.5.0.324). The package of MATLAB m-files, Matpower is being use for the large scale power systems simulation. Matpower is a package of MATLAB *m-files* which is used for solving power flow problems. MATLAB programming of power flow analysis is applied to compute the line currents, voltage drop for every branches, bus voltages and power losses for the radial distribution networks.

3.3 Newton-Raphson Power Flow Methodology

The fundamentals of power flow and studies of various power flow methods are significant for development of this project. Later, a suitable power flow method will be applied and develop by using MATLAB for radial distribution networks. The basic methods for solving power flow, Newton-Raphson method is found to be more efficient and practical for large power systems.

The number of iterations required to obtain a solution is independent of the system size, but more functional evaluations are required for each of the iteration. Hence, Newton-Raphson method is chosen for conventional power flow method. The Newton-Raphson power flow method solution is shown in Figure 2.

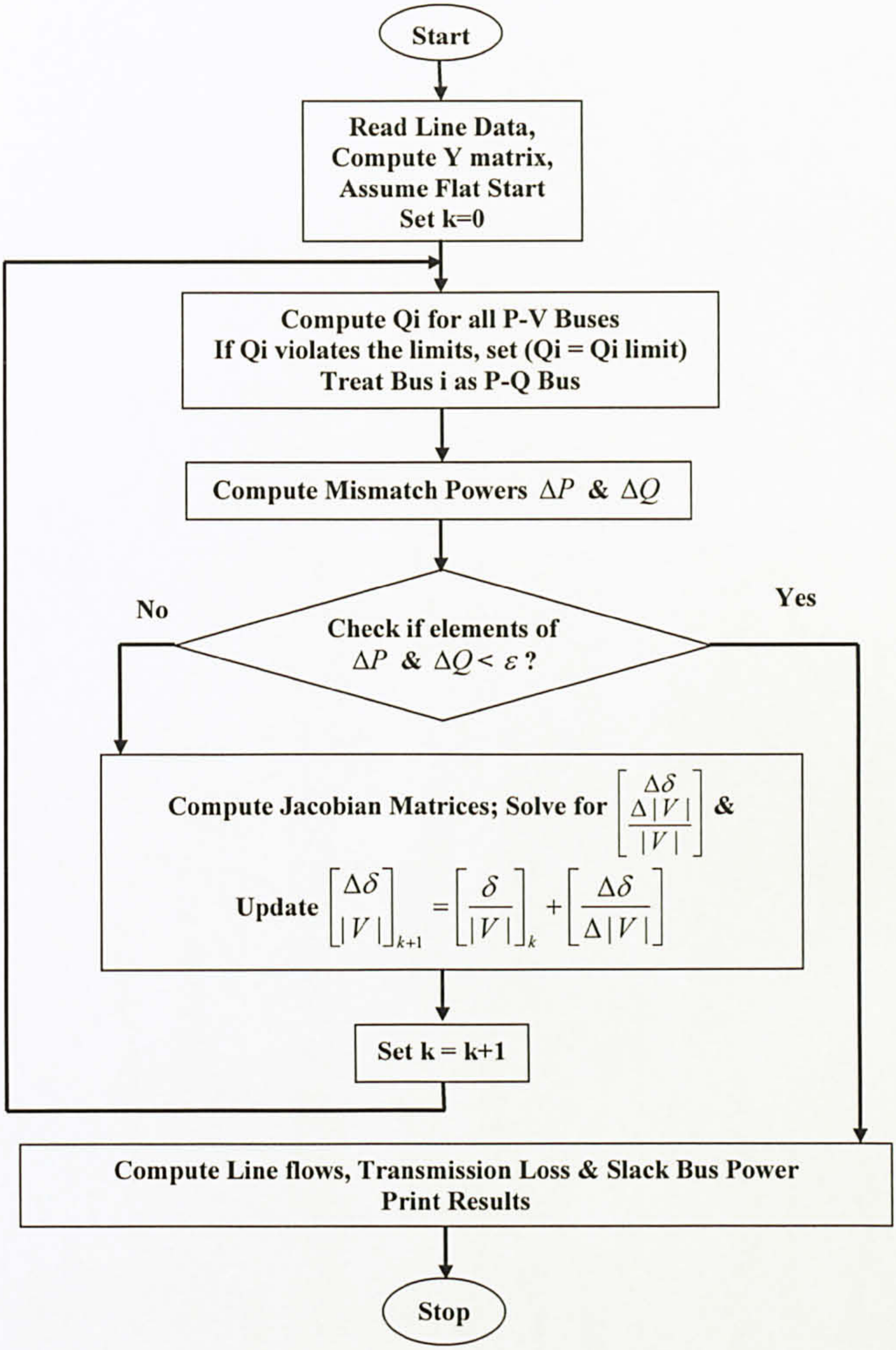

Figure 2: Newton-Raphson Power Flow Method Solution Flow Chart

The procedure of the Newton-Raphson method is generally as follows:

1. First, the line data and bus data are read. Then, admittance matrix (Y-impedance) will be computed.

2. Iteration count, k is set to 0. An initial guess of all unknown voltage magnitudes and angles are made. By assuming a flat start, all the voltage magnitudes are set to one while all the unknown voltage angles are set to zero.

3. Reactive power for all P-V buses, Q_i is computed. The reactive power limits for all generator buses are checked. If Qi for P-V bus is not in the range of the limits set $Q_i = Q_{i\ limit}$, then the bus *i* will change to type P-Q bus.

4. Mismatch powers (error vector) ΔP and ΔQ will be computed, which is the difference of the specified and the calculated power for both real power and reactive power. If the elements of error vector are less than the specified tolerance, ε then the problem is solved. Thus, go to Step 8; otherwise proceed to Step 5.

5. The elements of sub-matrices H,N,M and L are computed. Solve:

$$\begin{bmatrix} H & N \\ M & L \end{bmatrix} \begin{bmatrix} \delta \\ \dfrac{\Delta|V|}{V} \end{bmatrix} = \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \quad \textbf{for} \quad \begin{bmatrix} \delta \\ \dfrac{\Delta|V|}{V} \end{bmatrix}$$

6. The solution is updated as:

$$\begin{bmatrix} \delta \\ \dfrac{|V|}{V} \end{bmatrix}_{k+1} = \begin{bmatrix} \delta \\ |V| \end{bmatrix}_k + \begin{bmatrix} \Delta\delta \\ \Delta|V| \end{bmatrix}$$

7. K is then set to $k=k+1$ and return to Step 3.

8. Line flows, transmission loss and slack bus power are calculated. The results are printed.

Newton-Raphson power flow method is converted into MATLAB programming for efficiency, reliability and to save time. Newton-Raphson power flow solution by using MATLAB consist of several segments which include computation of admittance matrix (Y-bus), power generation, reactive power limits checking (Q limit), convergence, mismatch power calculation and computation of Jacobian matrix and updating state variables.

The default power flow solver is standard Newton-Raphson method by using full Jacobian with an update at each of the iterations. MATLAB programming by using Newton-Raphson power flow method to solve radial distribution networks is shown in Appendix II. It consists of power generations, reactive power limits checking, mismatch powers, Jacobian matrix formation and convergence.

3.4 Assumption

The three phase radial distribution networks are assumed to be balanced and it can be represented by their equivalent single-line diagrams. The line shunt capacitors are negligible in this case.

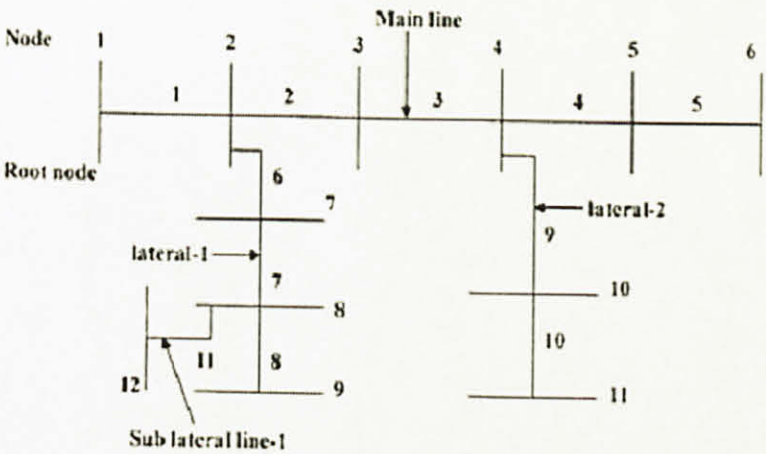

Figure 3: Single-line diagram of a 12-bus radial distribution network

3.5 PROPOSED SOLUTION APPROACH

Figure 3 shows an example of a distribution feeder's single-line diagram. In a balanced radial distribution networks, PV bus does not exist. The proposed solution consists of connection matrix, branch-node matrix, load currents calculation, voltage drop, line flows, line losses and computational time.

3.5.1 Connection Matrix, C

The number of branches is always one lesser than the number of buses (also known as nodes) in radial distribution network. Figure 4 shows the single-line diagram of 15-bus [3] distribution network.

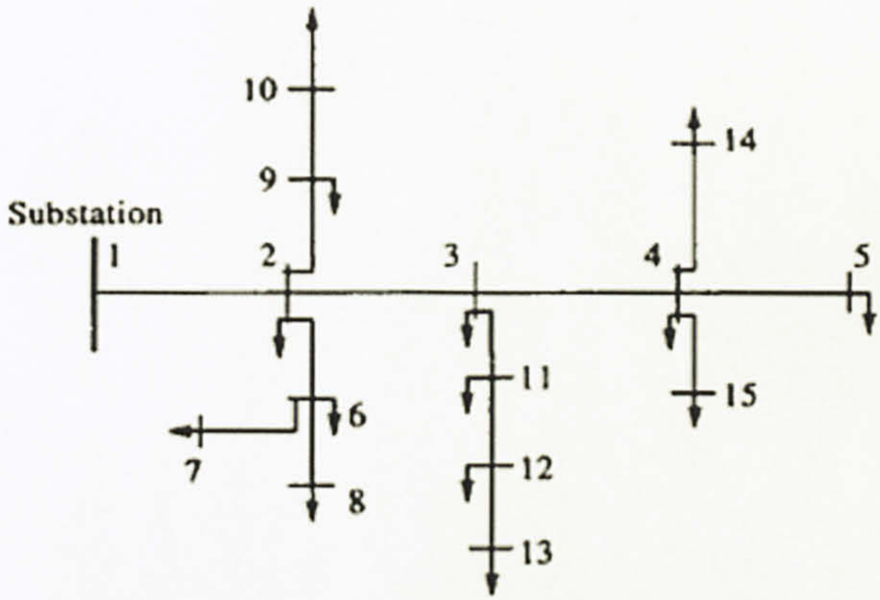

Figure 4: Single-line diagram of a 15-bus radial distribution network

The sending-ends and receiving-ends of the network branches of 15-bus network are listed in Table 1.

3.5 PROPOSED SOLUTION APPROACH

Figure 3 shows an example of a distribution feeder's single-line diagram. In a balanced radial distribution networks, PV bus does not exist. The proposed solution consists of connection matrix, branch-node matrix, load currents calculation, voltage drop, line flows, line losses and computational time.

3.5.1 Connection Matrix, C

The number of branches is always one lesser than the number of buses (also known as nodes) in radial distribution network. Figure 4 shows the single-line diagram of 15-bus [3] distribution network.

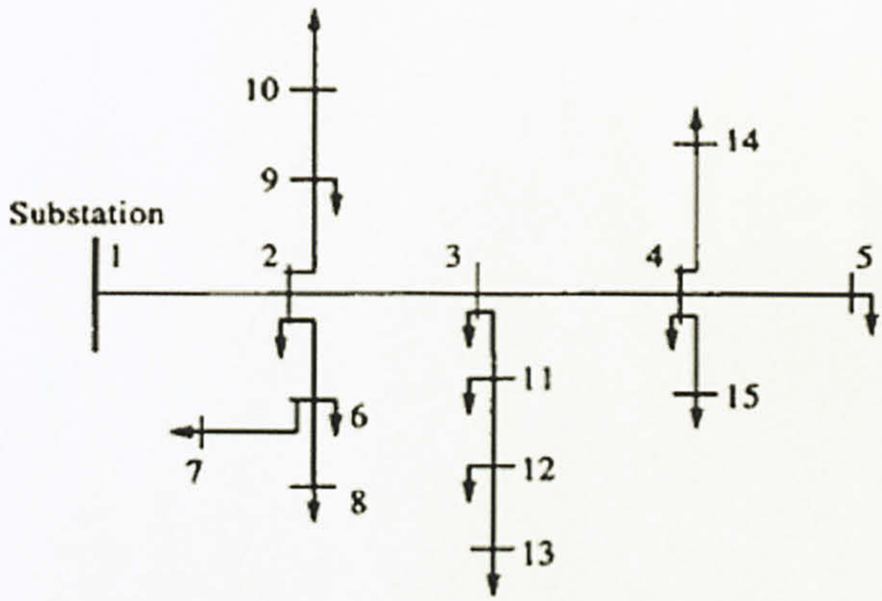

Figure 4: Single-line diagram of a 15-bus radial distribution network

The sending-ends and receiving-ends of the network branches of 15-bus network are listed in Table 1.

Table 1: Branch number, sending-end and receiving-end nodes.

Branch no.	1	2	3	4	5	6	7	8	9	10	11	12	14	15
Sending-end	1	2	3	4	2	9	2	6	6	3	11	12	4	4
Receiving-end	2	3	4	5	9	10	6	7	8	11	12	13	14	15

The connection matrix, **C** is a matrix which indicates the connection of the network branches. Each row of **C** matrix corresponds to a node. The elements in a row show the immediate nodes beyond.

Connection matrix is created by simply scanning through the sending-ends and receiving-ends of the branches one-by-one. The row number of **C** matrix is given by the sending-end node while the receiving-end node is filled in the corresponding row. The connection matrix, **C** for 15-bus system is shown in Figure 5.

$$
\mathbf{C} = \begin{array}{r|c|c|c|}
1 & 2 & & \\
2 & 3 & 6 & 9 \\
3 & 4 & 11 & \\
4 & 5 & 14 & 15 \\
5 & & & \\
6 & 7 & 8 & \\
7 & & & \\
8 & & & \\
9 & 10 & & \\
10 & & & \\
11 & 12 & & \\
12 & 13 & & \\
13 & & & \\
14 & & & \\
15 & & & \\
\end{array}
$$

Figure 5: Connection Matrix, **C** for 15-bus system

It is to be noted that there are a few terminating nodes, which are nodes 5, 7, 8, 10, 13, 14 and15. At terminating node, there is no any immediate node beyond it. Thus, the row in **C** matrix which is corresponding to a terminal node will be empty. In the subsequent calculations, the nodes beyond a particular branch are required. This can be obtained as discussed in the next section.

3.5.2 Branch-Node Matrix

Branch-node matrix, **BN** shows all the nodes beyond any given branch. Each row of **BN** matrix corresponds to one branch and all the nodes beyond that particular branch are indicated in the different columns in that row.

The following procedure is adopted to find the entries in BN matrix. All the branches are considered one-by-one. For any branch, its sending-end will be its first immediate node beyond, which is marked in the first column. The immediate nodes beyond it are obtained from the **C** matrix and entered in the **BN** matrix for each of the entry made in the **BN** matrix. This process is continued until corresponding to entries of **BN** matrix, no immediate nodes beyond them are obtained from the **C** matrix.

For the 15-bus network, the row corresponding to branch 3 is obtained as follows:

1. Firstly, the sending-end of branch 3 is {4}, which forms the first entry in row 3.
2. From **C** matrix, it can be seen that the immediate nodes beyond node 4 are {5 14 15} and these are entered in row 3.
3. Immediate nodes beyond node 5, 14 and 15 do not exist as they are terminal nodes.
4. Therefore, the row 3 of **BN** matrix is {4 5 14 15}.

The BN matrix for 15-bus network is shown in Figure 6.

Branch No.	Nodes beyond branch													
1	2	3	9	6	4	11	10	7	8	5	14	15	12	13
2	3	4	11	5	14	15	12	13	0	0	0	0	0	0
3	4	5	14	15	0	0	0	0	0	0	0	0	0	0
4	5	0	0	0	0	0	0	0	0	0	0	0	0	0
5	9	10	0	0	0	0	0	0	0	0	0	0	0	0
6	10	0	0	0	0	0	0	0	0	0	0	0	0	0
BN = *7*	6	7	8	0	0	0	0	0	0	0	0	0	0	0
8	7	0	0	0	0	0	0	0	0	0	0	0	0	0
9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
13	14	0	0	0	0	0	0	0	0	0	0	0	0	0
14	15	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6: Branch-Node Matrix, **BN** for 15-bus system

Flow chart of the proposed solution is shown in Figure 7 and the details of the approach are being discussed in the next section.

3.5.3 Proposed Solution Algorithm for Radial Distribution Networks

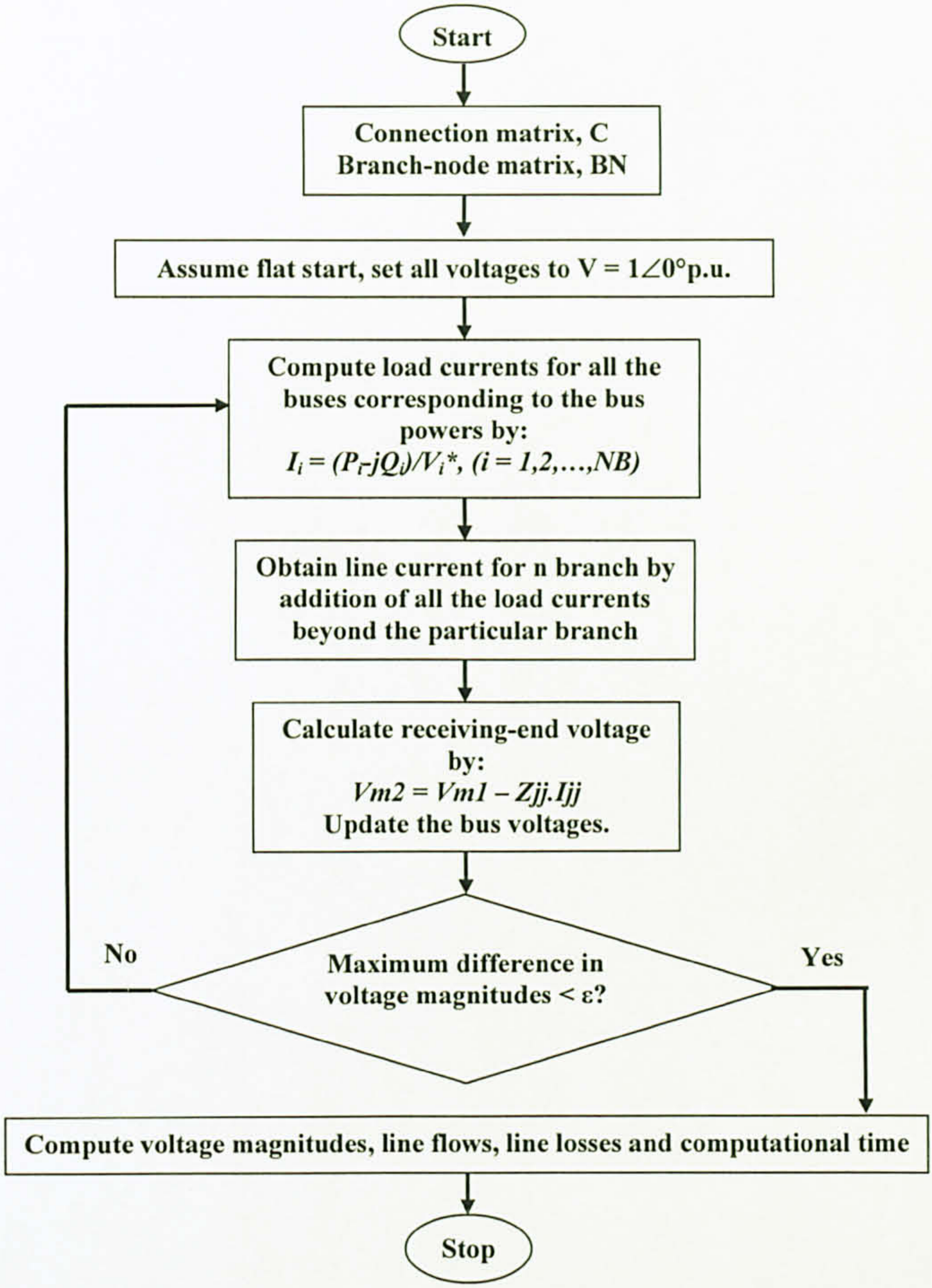

Figure 7: Flow Chart for Radial Distribution Network

The system configuration of the radial distribution is identified, which include the number of bus, from bus (sending-end node) and to bus (receiving-end node). From the total number of buses, the total number of branches (elements) can be determined whereas total number of branch is always one less than the total number of buses. The line data provided include the complex power, $S = P + jQ$ for every bus and line impedance for each and every branch.

The algorithm for computing updated node voltages by using the proposed method is as follows:

Step 1 Knowing the network data, obtain the connection matrix and hence the branch-node matrix BN .

Step 2 Assume flat start for node voltages, set voltages to V = 1+j0 p.u.

Step 3 Compute load currents for all buses corresponding to bus powers by using the complex power formula, $Si = Vi.\ I_L i^*$ whereby:

$$I_L i = \frac{P_L i - jQ_L i}{Vi^*};\ i=1,2,3...,NB$$

Step 4 Compute line currents for all the branches by the addition of all load currents beyond a particular branch.

Step 5 Compute voltage drop, Zjj.Ijj. Thus, calculate receiving-end voltage by:

$$Vm2 = Vm1 - Zjj.Ijj$$

where: jj = branch(element) number

m1 = IS(jj) = sending-end node

m2 = IR(jj) = receiving-end node

Step 6 Substitute the updated values of bus voltages for next iteration.

Step 7 Continue to iterate until the convergence is reached, whereby the bus voltages solution obtained is less than the tolerance, ε.

CHAPTER 4
RESULTS AND DISCUSSION

4.1 Results

The efficiency of the proposed approach has been tested by using the line data and load data from 15-bus [3], 28-bus [4], and 30-bus [17] systems. The line data and load data of the respective systems are shown in Appendix IV. Test results by MATLAB simulation for 15-bus, 28-bus and 30-bus systems are shown in Table 2, 3 and 4. Appendix II and Appendix III shows both MATLAB programming developed by using Newton-Raphson power flow method and proposed approach suited for radial distribution networks respectively.

4.1.1 Voltage Solution

Table 2: Load flow solution of 15-bus

Node No.	|V| (p.u.)	V angle (°)	Node No.	|V| (p.u.)	V angle (°)
1	1	0	9	0.968	0.072
2	0.9713	0.032	10	0.9669	0.085
3	0.9567	0.0493	11	0.95	0.1315
4	0.9509	0.0565	12	0.9458	0.1824
5	0.9499	0.0687	13	0.9445	0.1987
6	0.9582	0.1894	14	0.9486	0.0849
7	0.956	0.2166	15	0.9484	0.0869
8	0.957	0.205			

Table 3: Load flow solution of 28-bus

Node No.	|V| (p.u.)	V angle (°)	Node No.	|V| (p.u.)	V angle (°)
1	1	0	15	0.94995	4.3456
2	0.9505	0.24	16	0.04583	4.4346
3	0.8982	0.462	17	0.94452	4.7102
4	0.8701	0.6022	18	0.94861	4.7757
5	0.8521	0.6891	19	0.94844	0.3502
6	0.785	1.0099	20	0.9418	0.3798
7	0.7418	1.2271	21	0.9393	0.4161
8	0.7205	1.35	22	0.9373	0.4393
9	0.6839	1.5933	23	0.8937	0.5354
10	0.6396	1.8927	24	0.8902	0.5696
11	0.6117	2.4084	25	0.8867	0.6128
12	0.5996	2.6349	26	0.7846	1.0654
13	0.5771	3.2663	27	0.7833	1.084
14	0.5538	3.9793	28	0.7826	1.0933

Table 4: Load flow solution of 30-bus

Node No.	|V| (p.u.)	V angle (°)	Node No.	|V| (p.u.)	V angle (°)	Node No.	|V| (p.u.)	V angle (°)
1	1	0	11	0.922	1.2569	21	0.904	1.4902
2	0.9885	0.095	12	0.9217	1.2631	22	0.8974	1.6564
3	0.9785	0.1793	13	0.9779	0.1917	23	0.8919	1.7948
4	0.9646	0.3999	14	0.9775	0.203	24	0.8868	1.9263
5	0.9526	0.5949	15	0.9773	0.206	25	0.8843	1.9901
6	0.9413	0.7834	16	0.9773	0.2061	26	0.8834	2.0147
7	0.9338	0.9685	17	0.9323	0.9339	27	0.8831	2.022
8	0.9298	1.0656	18	0.9248	1.0612	28	0.933	0.9888
9	0.9251	1.1802	19	0.9165	1.2303	29	0.9324	1.0025
10	0.9229	1.2354	20	0.9099	1.3685	30	0.9322	1.0093

Table 5: Comparison of Newton-Raphson power flow method with proposed approach by number of iteration and computational time.

System	Power Flow Method	Iteration No.	CPU time (s)
15	Newton-Raphson	3	0.1109
	Proposed approach	6	0.0750
28	Newton-Raphson	6	0.2016
	Proposed approach	30	0.0999
30	Newton-Raphson	4	0.1297
	Proposed approach	5	0.0719

4.2 Discussion

The proposed approach have been implemented by using MATLAB R2008a (Version 7.6.0.324) and tested on a Intel® Pentium® M Processor 1.73Ghz 795MHz 512MB RAM computer. The memory usage by MATLAB on this computer is 259MB RAM, which is small.

4.2.1 Load Flow Solution Analysis

The test simulation results acquired for load flow solution by using both Newton-Raphson power flow method and the proposed approach for all the tested three systems (15-bus, 28-bus and 30-bus) are equivalent. The differences between the two methods are such as number of iteration and computational time to reach convergence.

The load flow results for the 15-bus and 30-bus acquired (shown in Table 2 and Table 4) are compatible with the results in [3] and [17]. However, the load flow result acquired for 28-bus (Table 3) differs from the result available in [4], which is shown in Figure 8.

Table 6: Load flow solution of 28-Bus from [4]

Node number	Voltage magnitudes (p.u.)	Node number	Voltage magnitudes (p.u.)
1	1.00000	15	0.88945
2	0.95884	16	0.88560
3	0.91877	17	0.88296
4	0.89598	18	0.87556
5	0.83736	19	0.86914
6	0.85068	20	0.86378
7	0.82855	21	0.86240
8	0.82189	22	0.86120
9	0.81233	23	0.81710
10	0.80821	24	0.81079
11	0.94989	25	0.80725
12	0.94655	26	0.80467
13	0.93980	27	0.80884
14	0.93684	28	0.80774

Generally, radial distribution networks are inductive load. Thus, the voltage magnitudes:

- of the subsequent nodes of the same feeder/line must be getting lesser and lesser than the voltage magnitudes of the previous nodes,
- of the nodes at the laterals must be lesser than the voltage magnitudes at the main feeder and
- of the nodes at the sub-laterals must be lesser than the voltage magnitude of the nodes at the laterals.

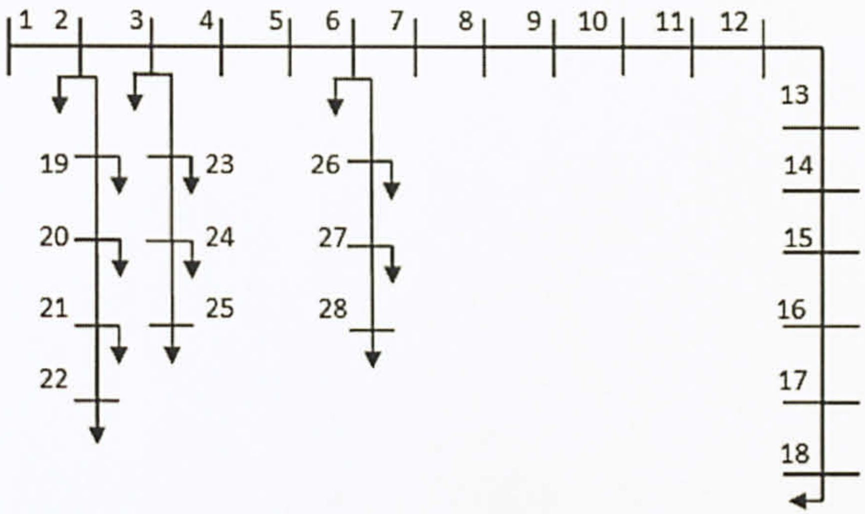

Figure 8: Single line diagram of 28-bus radial distribution network

From the result from [4] (shown in Table 6), the voltages solution is not compatible with the system configuration. It can be seen that the voltage magnitudes at node 6 is greater than node 5 and voltage magnitudes at node 11 is greater than voltage magnitudes at node 10. Such result is wrong as voltage magnitudes at node 6 and node 11 should be lesser than the voltage magnitudes at node 5 and node 10 respectively as they are of the same line/feeder.

In addition, it can be observed from Table 6 that the voltage magnitudes of node 27 and node 28 are greater than voltage magnitude of node 26. The voltage magnitudes of node 27 and 28 must be lesser than voltage magnitude of node 26 as they are of the same lateral. Hence, the result obtained in [4] is absolutely wrong.

Table 3 shows the accurate result of the 28-bus, which is tested by using the proposed method and Newton-Raphson method. It can be observed that the result obtained is quite consistent.

4.2.2 Comparison of Proposed Approach and Newton-Raphson Power Flow Method

From Table 5, it can be clearly seen that the proposed approach is significantly more efficient than the conventional power flow method because of fast convergence (less computational time). Although the proposed approach took more number of iterations to reach convergence, its computational time is much smaller as compared to Newton-Raphson power flow method.

Newton-Raphson took longer computational time because it involves computation of Jacobian matrix, inverse Jacobian and trigonometric equations. Thus, this method is more time consuming as it involves complex equations. Meanwhile, the present work involves only simple power flow equations, such as complex power formula and voltage drop formula and it does not involve any complex equations.

Besides, Newton-Raphson power flow method took longer time to converge because the programme is large as it has to call several different functions, such as calculating powers, check reactive power limits and compute mismatch powers. Meanwhile the proposed approach only calls one function, which is to check null vector.

4.2.3 Proposed Approach

Radial distribution networks differ from other types of distribution networks by its simple radial structure (open path), which generally cause higher R/X ratio. Thus, special programme is designed to suit the radial distribution networks. Newton-Raphson power flow method and power flow software such as ERACS are much powerful methods and hence longer computational time. They are more suitable to solve for complex distribution networks, such as interconnected distribution networks (closed path) rather than radial distribution networks (open path).

Moreover, Jacobian matrix may not be singular matrix for radial distribution networks with high R/X ratios, which may result the systems in ill-conditioned. Furthermore, it is not necessary to apply conventional power flow method and power flow software to solve radial distribution networks since power flow is not necessary to solve the radial distribution networks.

The proposed approach is a much simpler method as there is no need to compute Jacobian matrix and it involves only simple power flow calculations. Besides, it applies much simpler algorithm (Figure 7) which is specially designed to suit and solve radial distribution networks.

The algorithm applied in the present method is much simpler and less complicated as compared to the complex algorithms which are used in [3] and [4]. The algorithms presented in [3] and [4] to construct nodes and branches beyond a particular node/nodes beyond all the branches are given in Appendix V. It can be seen that these algorithms are much more complex, which involve several logical decisions.

Meanwhile, the proposed method is much simpler by using the concept of connection matrix, **C** and branch-node matrix, **BN**. The computation of **C** and **BN** are explained in Section 3.5 earlier on.

4.2.4 Accommodating Voltage Dependent Loads

In normal power flow studies, the value of load powers remains constant regardless of the system frequency and node voltage magnitudes. However, in any practical system, combination of several types of loads such as residential, industrial and commercial might be present. In this case, the real and reactive powers of such loads depend on node voltages and can be expressed as in [17]:

$$P = P_o\left(\frac{V}{V_o}\right)^{np}$$

$$Q = Q_o \left(\frac{V}{V_o} \right)^{nq}$$

where n_p and n_q stand for load exponents; P_o and Q_o are real and reactive powers corresponding to voltage Vo; P and Q are corresponding to voltage V.

Values of np and nq depend on the type of loads, which is shown in Table 7. Each iteration load powers can be updated by using the updated node voltage by knowing the fraction of each component of the load. By this, it is possible to incorporate voltage dependent loads in the present work for any practical data if the different types of load components are specified.

Table 7: Common Values for the Exponents for Different Static Load Models

Load component	n_p	n_q
Battery charge	2.59	4.06
Fluorescent lamps	2.07	3.21
Constant impedance	2	2
Fluorescent lighting	1	3
Air conditioner	0.5	2.5
Constant current	1	1
Resistance space heater	2	0
Pumps, funs other motors	0.08	1.6
Incandescent lamps	1.54	0
Compact fluorescent lamps	1	0.35
Small industrial motors	0.1	0.6
Large industrial motors	0.05	0.5
Constant power	0	0

CHAPTER 5
CONCLUSION AND RECOMMENDATION

5.1 Conclusion

A suitable power flow method which is specially designed to solve radial distribution networks which is simple, efficient and practical is successfully acquired and developed in MATLAB. It uses much simpler algorithm to construct connection matrix, **C** and branch-node matrix, **BN**. The MATLAB programming developed was tested on several standard radial distribution networks and from the test, it is discovered that the proposed approach is efficient as the computational time is small. The validity of the results obtained by proposed approach is strengthen by Newton-Raphson method as the voltage solutions acquired by using both methods are equal.

5.2 Recommendation

The proposed approach for radial distribution network can be implemented on any practical data. In case if there is any convergence problem, (especially for larger radial distribution systems) the developed method can be modified incorporating forward and backward voltage updating as suggested in reference number [17].

REFERENCE

[1] Hadi Saadat. 2002, Power System Analysis, United States, McGraw-Hill

[2] K.N. Hasan, K.S.R. Rao, Z. Mokhtar, 2008. "Analysis of Load Flow and Short Circuit Studies of an Offshore Platform Using ERACS Software"

[3] D. Das, D. P. Kothari, A Kalam, 1995, "Simple and efficient method for load flow solution of radial distribution networks"

[4] S. Ghosh and D.Das, 1999, "Method for load-flow solution of radial distribution networks".

[5] Kersting, W.H., and Mendive, D.L., 1976, "An application of ladder network theory to the solution of three phase radial load flow problem"

[6] Kersting, W.H., 1984, "A method to the design and operation of a distribution system"

[7] Stevens, R.A., Rizy, D.T., and Purucker, S.L., 1986 "Performance of conventional power flow routines for real-time distribution automation application" in *System theory*, 18th Southeastern symposium; IEEE Computer Society

[8] Shirmohammadi, D., Hong, H.W., Semlyen, A., and Luo, G.X., 1988, "A compensation-based power flow method for weakly meshed distribution and transmission networks"

[9] 27th May 2008 <http://en.wikipedia.org/wiki/Power_flow_analysis>

[10] Baran, M.E., and Wu, F.F., 1989, "Optimal sizing of capacitors placed on a radial distribution system"

[11] Chiang, H.D., 1991, "A decoupled load flow method for distribution power network: algorithms, analysis and convergence study"

[12] Renato, C.G., 1990, "New method for the analysis of distribution networks"

[13] Goswami, S.K., and Basu, S.K., 1991, "Direct solution of distribution systems"

[14] Jasmon, G.B., and Lee, L.H.C.C., 1991, "Stability of load flow techniques for distribution system voltage stability analysis"

[15] Jasmon, G.B., and Lee, L.H.C.C., 1991, "Distribution network reduction for voltage stability analysis and load flow calculations"

[16] J. J. Grainger and W. D. Stevenson. 1994, Power System Analysis, New York, McGraw-Hill

[17] Ulas Eminoglu, M.Hakan Hocaoglu, 2005, "A New Power Flow Method for Radial Distribution Systems Including Voltage Dependent Load Models"

[18] S.F. Mekhamer, S.A. Soliman, M.A. Mostafa, M.E. El-Hawary, 2001, "Load Flow Solution of Radial Distribution Feeders: A New Approach"

APPENDICES

APPENDIX I: PROJECT GANTT CHART

Table 8: Final Year Project – Semester I

No.	Detail / Week	1	2	3	4	5	6	7	8	9		10	11	12	13	14
1	Project Topic Selection and Proposal										Mid-Semester Break					
2	Submission on Project Proposal															
3	Project Research															
4	Submission of Preliminary Report															
5	Power Flow Studies															
6	MATLAB Tutorial - Power Flow Solution by using Newton Raphson															
7	Submission of Progress Report															
9	Studies on Radial Distribution Networks															
10	Submission on Draft Report (Interim)															
11	Seminar															
12	Submission of Interim Report															
13	Oral Presentation															

Table 9: Final Year Project – Semester 2

No.	Detail / Week	1	2	3	4	5	6	7	8	9		10	11	12	13	14	15	16	17	18	19
1	Matlab Development										Mid-Semester Break										
2	Submission of Progress Report 1																				
3	Matlab Case Studies																				
4	Submission of Progress Report 2																				
5	Seminar																				
6	Poster Presentation (Pre-EDX)																				
6	Submission of Draft Report																				
7	Submission of Final Report (Soft Cover)																				
8	Submission on Technical Report																				
9	Oral Presentations																				
10	Submission of Final Report (Hard Cover)																				

APPENDIX II: NEWTON-RAPHSON POWER FLOW SOLUTION FOR RADIAL DISTRIBUTION NETWORKS

Main Code:

```
% Power Flow Solution of Radial Distribution Network - 15 Bus -
%
clear all;
clc;
tstart=cputime;
disp(['***NR Power Flow Solution of Radial Distribution Network - 15
Bus***'])
Nbus = 15; Nele = 14; Nsh = 0;
%
% Reading element data
%
Edata = [1 1 2 1.35309+1.32349i 0 1;2 2 3 1.17024+1.14464i 0 1;3 3 4
         0.84111+0.82271i 0 1;...
         4 4 5 1.52348+1.02760i 0 1;5 2 9 2.01317+1.35790i 0 1;6 9 10
         1.68671+1.13770i 0 1;...
         7 2 6 2.55727+1.72490i 0 1;8 6 7 1.08820+0.73400i 0 1;9 6 8
         1.25143+0.84410i 0 1;...
         10 3 11 1.79553+1.21110i 0 1;11 11 12 2.44845+1.65150i 0 1;12
         12 13 2.01317+1.35790i 0 1;...
         13 4 14 2.23081+1.50470i 0 1;14 4 15 1.19702+0.80740i 0 1];
%
% Reading shunt data
%
Shdata = [0 0 0];
%
% Reading bus data
%
Bdata = [1 1 0.0;2 3 63.0;3 3 100.0;4 3 200.0;5 3 63.0;...
         6 3 200.0;7 3 200.0;8 3 100.0;9 3 100.0;10 3 63.0;...
         11 3 200.0;12 3 100.0;13 3 63.0;14 3 100.0;15 3 200.0];
%
% Displaying data
%
disp(['    Sl.No              From bus              To bus
           Line Impedance            HLCA             ONTR'])
Edata
if Nsh~=0
    disp(['   Sl.No.               At bus              Shunt Admittamce'])
    Shdata
end
disp(['   Sl.No. Bus Type   KVA'])
Bdata
%
% Formation of Ybus matrix
%
Ybus = zeros(Nbus,Nbus);
for k = 1:Nele
```

```
        p = Edata(k,2);
        q = Edata(k,3);
        imp = Edata(k,4)*100/(11*11);
        yele = 1/imp;
        Hlca = Edata(k,5);
        offa = Edata(k,6);
        offaa = offa*offa;
        Ybus(p,p) = Ybus(p,p) + yele/offaa + Hlca;
        Ybus(q,q) = Ybus(q,q) + yele + Hlca;
        Ybus(p,q) = Ybus(p,q) - yele/offa;
        Ybus(q,p) = Ybus(q,p) - yele/offa;
    end
if Nsh~=0
    for i = 1:Nsh
        q = Shdata(i,2);
        yele = Shdata(i,3);
        Ybus(q,q) = Ybus(q,q) + yele;
    end
end
Ymag=abs(Ybus);
Yang=angle(Ybus);
for i=1:Nbus
    Gii(i)=real(Ybus(i,i));
    Bii(i)=imag(Ybus(i,i));
end
%
% Transferring bus data
%
NNbus = 2*Nbus;
for i = 1:Nbus
    Bustype(i) = Bdata(i,2);
    Pgen(i) = 0;
    Qgen(i) = 0;
    KVA = Bdata(i,3);
    Pload(i) = KVA*0.7/100000;
    sintheta = sin(acos(0.7));
    Qload(i) = KVA*sintheta/100000;
    VM(i) = 1.0;
    VA(i) = 0.0;
    Qmin(i) = 0.0;
    Qmax(i) = 10.0;
end
%
% Formation of pvec and qvec
%
pvec = ones(1,Nbus);
qvec = ones(1,Nbus);
for ii = 1:Nbus
    if Bustype(ii)==1
        Nslack = ii;
        pvec(ii) = 0;
        qvec(ii) = 0;
    end
end
for ii = 1:Nbus
    if Bustype(ii)==2
            qvec(ii) = 0;
```

```
    end
end
%
% General settings
%
flag = 0;
iter = 0;
itmax = 5;
tol = 1e-6;
itmax
tol
%
% Calculation of net powers
%
for i = 1:Nbus
    Pnet(i) = Pgen(i)-Pload(i);
    Qnet(i) = Qgen(i)-Qload(i);
end
%
% Iteration starts
%
while( iter < itmax & flag == 0)
    [Pcal,Qcal] = CalcuPowers_Grain(Nbus,VM,VA,Ymag,Yang);
    [Qnet,Bustype,flag2] =
QlimitChecking(Nbus,Bustype,Qgen,Qload,Qmin,Qmax,Qnet,Qcal,iter);
    if flag2==1
        qvec = ones(1,Nbus);
        qvec(Nslack) = 0;
        for ii = 1:Nbus
            if Bustype(ii)==2
                qvec(ii) = 0
            end
        end
    else
    end
    [DPQ,DP,DQ,flag,maxerror] =
MismatchPowers(Nbus,NNbus,tol,pvec,qvec,flag,Pnet,Qnet,Pcal,Qcal);
    iter = iter +1;
    [JAC] =
Jacobi_Grain(Nbus,NNbus,pvec,qvec,VM,VA,Ymag,Yang,Pcal,Qcal,Gii,Bii);
    Delta = JAC\DPQ';
    [VM,VA] = UpdatingState(Nbus,Delta,VM,VA,pvec,qvec);
end
iter     % Iteration Number
maxerror     % Maximum error
VM     % Bus voltage magnitude
VA = VA*180/pi     % Bus voltage phase angle (Deg)
time=cputime-tstart
```

Functions Codes:

Powers Calculation

```
function [Pcal,Qcal] = CalcuPowers_Grain(Nbus,VM,VA,Ymag,Yang);
Pcal=zeros(Nbus,1);
Qcal=zeros(Nbus,1);
for i=1:Nbus
    for n=1:Nbus
        theta = Yang(i,n)+VA(n)-VA(i);
        Pcal(i) = Pcal(i)+Ymag(i,n)*VM(i)*VM(n)*cos(theta);
        Qcal(i) = Qcal(i)-Ymag(i,n)*VM(i)*VM(n)*sin(theta);
    end
end
```

Q limit Checking

```
function[Qnet,Bustype,flag2] =
QlimitChecking(Nbus,Bustype,Qgen,Qload,Qmin,Qmax,Qnet,Qcal,iter);
flag2 = 0;
if iter>2
    for ii = 1:Nbus
        if Bustype(ii)==2
            Qgen(ii) = Qcal(ii) + Qload(ii);
            if Qgen(ii)>Qmax(ii)
                Qgen(ii) = Qmax(ii);
                Qnet(ii) = Qgen(ii) - Qload(ii);
                Bustype(ii) = 3;
                flag2 = 1;
            else
                if Qgen(ii)<Qmin(ii)
                    Qgen(ii) = Qmin;
                    Qnet(ii) = Qgen(ii) - Qload(ii);
                    Bustype(ii) = 3;
                    flag2 = 1;
                end
            end
        end
    end
else
end
```

Mismatch Powers

```
function [DPQ,DP,DQ,flag,maxerror] =
MismatchPowers(Nbus,NNBus,tol,pvec,qvec,flag,Pnet,Qnet,Pcal,Qcal);
DPQ = zeros(1,NNbus);
DP = zeros(1,Nbus);
DQ = zeros (1,Nbus);
for ii = 1:Nbus
    if pvec(ii)==1
        DP(ii) = Pnet(ii) - Pcal(ii);
    end
    if qvec(ii)==1
        DQ(ii) = Qnet(ii) - Qcal(ii);
    end
end
for ii = 1:Nbus
    iii = ii + Nbus;
    DPQ(ii) = DP(ii);
    DPQ(iii) = DQ(ii);
end
%
%Check for convergence
%
maxerror = max(max(abs(DP)),max(abs(DQ)));
if maxerror<tol
    flag = 1;
end
```

Jacobian Matrix Formation

```
[JAC] =
Jacobi_Grain(Nbus,NNbus,pvec,qvec,VM,VA,Ymag,Yang,Pcal,Qcal,Gii,Bii);
JAC=zeros(NNbus,NNbus);
%
% H matrix
%
for i=1:Nbus
    for j=1:Nbus
        theta=Yang(i,j)+VA(j)-VA(i);
        if pvec(i)==1 & pvec(j)==1
            if i==j
                JAC(i,i)=-Qcal(i)-VM(i)*VM(i)*Bii(i);
            else
                JAC(i,j)=-VM(i)*VM(j)*Ymag(i,j)*sin(theta);
            end
        end
    end
end
for i=1:Nbus
    for j=1:Nbus
        theta=Yang(i,j)+VA(j)-VA(i);
        if qvec(i)==1 & pvec(j)==1
            ii = i+Nbus;
            if i==j
                JAC(ii,i)=Pcal(i)-VM(i)*VM(i)*Gii(i);
```

```
            else
                JAC(ii,j)=-VM(i)*VM(j)*Ymag(i,j)*cos(theta);
            end
        end
    end
end
for i=1:Nbus
    for j=1:Nbus
        theta=Yang(i,j)+VA(j)-VA(i);
        if pvec(i)==1 & qvec(j)==1
            jj = j+Nbus;
            if i==j
                JAC(i,jj)=Pcal(i)+VM(i)*VM(i)*Gii(i);
            else
                JAC(i,jj)=VM(i)*VM(j)*Ymag(i,j)*cos(theta);
            end
        end
    end
end
for i=1:Nbus
    for j=1:Nbus
        theta=Yang(i,j)+VA(j)-VA(i);
        if qvec(i)==1 & qvec(j)==1
            ii = i+Nbus;
            jj = j+Nbus;
            if i==j
                JAC(ii,jj)=Qcal(i)-VM(i)*VM(i)*Bii(i);
            else
                JAC(ii,jj)=-VM(i)*VM(j)*Ymag(i,j)*sin(theta);
            end
        end
    end
end
for i=1:NNbus
    if JAC(i,i)==0
        JAC(i,i)=1.0;
    end
end
```

Updating States

```
function[VM,VA] = UpdatingState(Nbus,Delta,VM,VA,pvec,qvec);
for ii = 1:Nbus
    if pvec(ii)==1
        VA(ii) = VA(ii) + Delta(ii);
    end
    iii = ii + Nbus;
    if qvec(ii)==1
        VM(ii) = VM(ii) + Delta(iii)*VM(ii);
    end
end
```

APPENDIX III: PROPOSED MATLAB PROGRAMMING FOR RADIAL DISTRIBUTION NETWORKS

15-bus System

```
% ***** RADIAL DISTRIBUTION NETWORKS - 15 BUS SYSTEM *****
clear all;
clc;
tstart = cputime;
disp (['***** RADIAL DISTRIBUTION NETWORKS - 15 BUS SYSTEM *****'])
Nbus = 15; BaseKV = 11; BaseMVA = 100;
%
% Reading element data (line data)
%
Nele = Nbus-1;
Edata = [1 1 2 1.35309+1.32349i; 2 2 3 1.17024+1.14464i; 3 3 4
         0.84111+0.82271i;...
         4 4 5 1.52348+1.02760i; 5 2 9 2.01317+1.35790i; 6 9 10
         1.68671+1.13770i;...
         7 2 6 2.55727+1.72490i; 8 6 7 1.08820+0.73400i; 9 6 8
         1.25143+0.84410i;...
         10 3 11 1.79553+1.21110i; 11 11 12 2.44845+1.65150i; 12 12 13
         2.01317+1.35790i;...
         13 4 14 2.23081+1.50470i; 14 4 15 1.19702+0.807401i];
%
% Reading bus data
%
Bdata = [1 0.0; 2 63.0; 3 100; 4 200.0; 5 63.0;...
         6 200.0; 7 200.0; 8 100.0; 9 100.0; 10 63.0;...
         11 200.0; 12 100.0; 13 63.0; 14 100.0; 15 200.0];
%
% Displaying data
%
disp(['    Sl.No.          From bus                To bus              Line
Impedance'])
Edata
disp(['   Sl.No.     KVA'])
Bdata
%
% Formation of connection matrix Cmat
%
Cmat = zeros (Nbus,5);
NCE = zeros (Nbus,1);
for jj = 1:Nele
    p = Edata(jj,2);
    q = Edata(jj,3);
    NCE(p) = NCE(p)+1;
    NCEp = NCE (p);
    Cmat(p,NCEp)=q;
end
Cmat
%
% Finding the nodes beyond element jj
%
NBEjj = zeros (Nele, Nele);
Temp = zeros (1,10);
TEMP = zeros (1,10);
%
% Set all first columns in NBEjj to mirror q
```

```
%
forCu rrRow = 1:Nele
    NBEjj(CurrRow, 1) = Edata(CurrRow, 3);
end
%
% Start iterating through whole NBEjj
%
forCu rrRow = 1:Nele
    CurrPos = 1; % Starting at col. 1, this is the position we're
checking
    for CurrCol = 1:Nele
        CurrQ = NBEjj(CurrRow,CurrCol);
        if CurrQ ~= 0
            vec = Cmat(CurrQ,:);
            [flag,Nonzero] = NULL(vec);
            if flag == 0 % If the connected vector is not Null
                for I = 1:Nonzero % Copy the nonzero values from
vector to NBEjj
                    NBEjj(CurrRow,CurrPos+I) = vec(I);  % Starting
from CurrPos+1
                end
                CurrPos = CurrPos + Nonzero; % Update CurrPos to
include added values
            end
        end
    end
end
NBEjj
%
% Calculation of line impedances
%
forI  = 1:Nele
    zij(I) = Edata(I,4)*BaseMVA/(BaseKV*BaseKV);
end
%
% Initializing bus voltages
%
forI  = 1:Nbus
    Vbus(I) = 1.0+0.0i;
    VBUS(I) = Vbus(I);
end
%
% Finding number of nodes beyond elements
%
forI  = 1:Nele
    NNODEjj(I) = 0;
    for J = 1:Nele
        if NBEjj(I,J)~=0
            NNODEjj(I) = NNODEjj(I)+1;
        end
    end
end
NNODEjj
%
% General settings
%
Delta = zeros(1,Nbus);
iter = 1;
itmax = 20;
tol = 1e-6;
ACF = 1.0;
itmax
tol
maxerror = 0.5;
%
```

```
% Iteration starts
%
Vbus
while(iter<itmax & maxerror>tol)
%
% Computing bus currents
%
    for I = 1:Nbus
        PL(I) = Bdata(I,2)*0.7/100000;
        sintheta = sin(acos(0.7));
        QL(I) = Bdata(I,2)*sintheta/100000;
        X = PL(I) - (j*QL(I));
        Y = Vbus(I);
        YY = conj(Y);
        IL(I) = X/YY;
    end
%
% Calculation of element currents
%
   for I = 1:Nele
       EleCurr(I) = 0;
       for J = 1:NNODEjj(I)
           NODE = NBEjj(I,J);
           EleCurr(I) = EleCurr(I) + IL(NODE);
       end
   end
       for JJ = 1:Nele
           m1 = Edata (JJ,2);
           m2 = Edata (JJ,3);
           Vbus(m2) = Vbus(m1) - EleCurr(JJ)*zij(JJ);
       end
       for I = 1:Nbus
           Vbus(I) = VBUS(I) + ACF*(Vbus(I) - VBUS(I));
       end
       for I = 1:Nbus
           Delta(I) = abs(abs(Vbus(I))-abs(VBUS(I)));
           VBUS(I) = Vbus(I);
       end
       maxerror = max(Delta);
       iter = iter + 1;
end
iter
maxerror
ACF
Vbus
for I = 1:Nbus
    VM(I) = abs (Vbus(I));
end
VM
disp(['Bus real powers'])
PL
disp(['Bus reactive powers'])
QL
disp(['Element currents'])
EleCurr
disp(['Element impedances'])
zij
disp(['Node currents'])
IL
for I = 1:Nele
    drop(I) = zij(I)*EleCurr(I);
end
disp (['Voltage drop in elements'])
drop
time = cputime - tstart
```

Function Code: Null Vector Checking

```
%
% Function to check NULL vector
%
function [flag,Nonzero] = NULL(vec)
flag = 1;
Nonzero = 0;
X = [1 1 1 1 1];
Y = X*vec';
if Y~=0
    flag = 0;
    Nonzero = 0;
    for j = 1:5
        if vec(j) ~= 0
            Nonzero = Nonzero+1;
        end
    end
end
```

APPENDIX IV: LINE DATA, BUS DATA AND LOAD DATA FOR RADIAL DISTRIBUTION NETWORKS

Table 10: Data for 15-bus Radial Distribution Network [3]

Branch number	Sending-end node	Receiving-end node	R (ohm)	X (ohm)
1	1	2	1.3531	1.3235
2	2	3	1.1702	1.1446
3	3	4	0.8411	0.8227
4	4	5	1.5235	1.0276
5	2	9	2.0132	1.3579
6	9	10	1.6867	1.1377
7	2	6	2.5573	1.7249
8	6	7	1.0882	0.734
9	6	8	1.2514	0.8441
10	3	11	1.7955	1.2111
11	11	12	2.4284	1.6515
12	12	13	2.0132	1.3579
13	4	14	2.2308	1.5074
14	4	15	1.197	0.8074

Nodes	KVA	Nodes	KVA
1	0	9	100
2	63	10	63
3	100	11	200
4	200	12	100
5	63	13	63
6	200	14	100
7	200	15	200
8	100		

11kV, 100 MVA

Power factor of the load: $\cos\phi = 0.70$

Real power load: $P = KVA*\cos\phi$

Reactive power load: $Q = KVA*\sin\phi$

Table 11: Data for 28-bus Radial Distribution Network [4]

Branch number	Sending-end node	Receiving-end node	R (ohm)	X (ohm)	P of receiving-end (kW)	Q of receiving-end (kVAr)
1	1	2	1.8216	0.7580	140.0	190.0
2	2	3	2.2270	0.9475	80.0	50.0
3	3	4	1.3662	0.5685	80.0	60.0
4	4	5	0.9180	0.3790	100.0	60.0
5	5	6	3.6432	1.5160	80.0	50.0
6	6	7	2.7324	1.1370	90.0	40.0
7	7	8	1.4573	0.6064	90.0	40.0
8	8	9	2.7324	1.1370	80.0	50.0
9	9	10	3.6432	1.5160	90.0	50.0
10	10	11	2.7520	0.7780	80.0	50.0
11	11	12	1.3760	0.3890	80.0	40.0
12	12	13	4.1280	1.1670	90.0	50.0
13	13	14	4.1280	0.8558	70.0	40.0
14	14	15	3.0272	0.7780	70.0	40.0
15	15	16	2.7520	1.1680	70.0	40.0
16	16	17	4.1280	0.7780	60.0	30.0
17	17	18	2.7520	0.7780	60.0	30.0
18	2	19	3.4400	0.9725	70.0	40.0
19	19	20	1.3760	0.3890	50.0	30.0
20	20	21	2.7520	0.7780	50.0	30.0
21	21	22	4.9536	1.4004	40.0	20.0
22	3	23	3.5776	1.0114	50.0	30.0
23	23	24	3.0272	0.8558	50.0	20.0
24	24	25	5.5040	1.5560	60.0	30.0
25	6	26	2.7520	0.7780	40.0	20.0
26	26	27	1.3760	0.3890	40.0	20.0
27	27	28	1.3760	0.3890	40.0	20.0

11kV, 100 MVA

Power factor of the load: $\cos\phi = 0.70$

Real power load: $P = KVA^*\cos\phi$

Reactive power load: $Q = KVA^*\sin\phi$

Table 12: Data for 30-bus Radial Distribution Network [17]

Branch number	Sending-end node	Receiving-end node	R (pu)	X (pu)	P of receiving-end (pu)	Q of receiving-end (pu)
1	1	2	0.0967	0.0397	0.0042	0.0026
2	2	3	0.0886	0.0364	0	0
3	3	4	0.1359	0.0377	0.0042	0.0026
4	4	5	0.1236	0.0343	0.0042	0.0026
5	5	6	0.1236	0.0343	0	0
6	6	7	0.2598	0.0446	0	0
7	7	8	0.1732	0.0298	0.0042	0.0026
8	8	9	0.2598	0.0446	0.0042	0.0026
9	9	10	0.1732	0.0298	0.0041	0.0025
10	10	11	0.1083	0.0186	0.0042	0.0026
11	11	12	0.0866	0.0149	0.0025	0.0015
12	3	13	0.1299	0.0233	0.0011	0.0007
13	13	14	0.1732	0.0298	0.0011	0.0007
14	14	15	0.0866	0.0149	0.0011	0.0007
15	15	16	0.0433	0.0074	0.0002	0.0001
16	6	17	0.1483	0.0412	0.0044	0.0027
17	17	18	0.1359	0.0377	0.0044	0.0027
18	18	19	0.1718	0.0391	0.0044	0.0027
19	19	20	0.1562	0.0355	0.0044	0.0027
20	20	21	0.1562	0.0355	0.0044	0.0027
21	21	22	0.2165	0.0372	0.0044	0.0027
22	22	23	0.2165	0.0372	0.0044	0.0027
23	23	24	0.2598	0.0446	0.0044	0.0027
24	24	25	0.1732	0.0298	0.0044	0.0027
25	25	26	0.1083	0.0186	0.0044	0.0027
26	26	27	0.0866	0.0149	0.0026	0.0016
27	7	28	0.1299	0.0223	0.0017	0.0011
28	28	29	0.1299	0.0223	0.0017	0.0011
29	29	30	0.1299	0.0223	0.0017	0.0011

APPENDIX V: RESEARCHER'S FLOW CHARTS

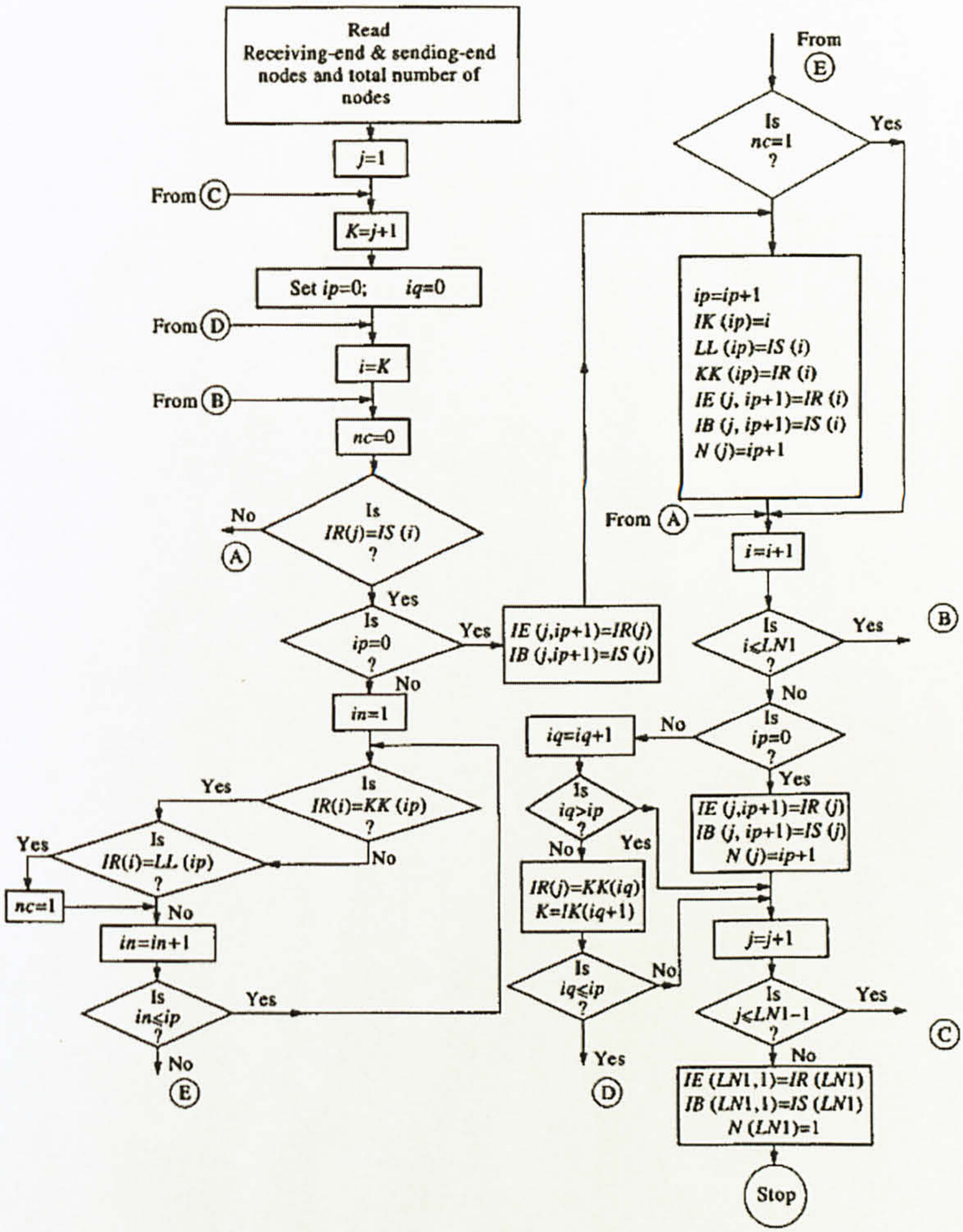

Figure 9: Flow Chart for Identification of Nodes and Branches beyond a Particular Node [3]

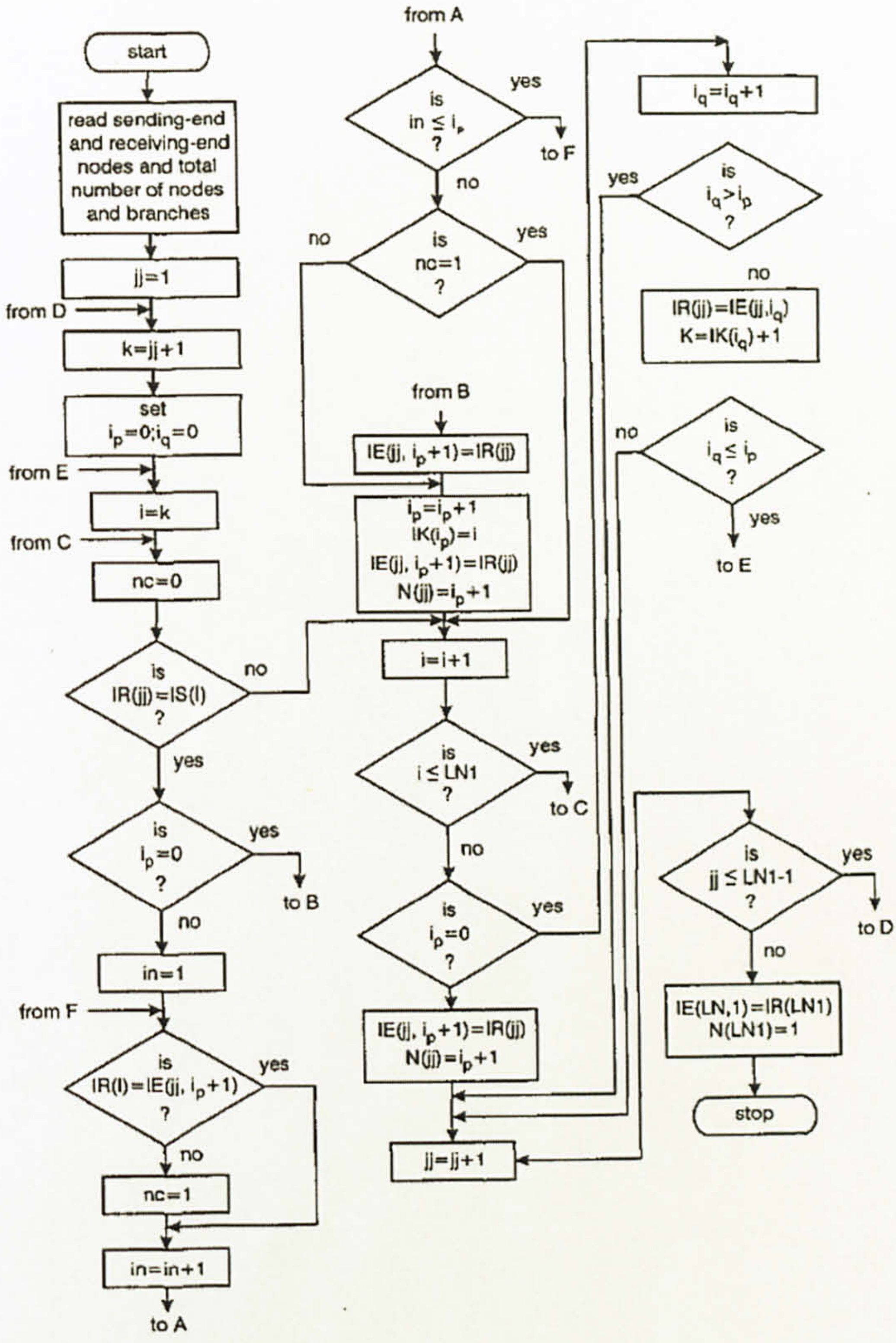

Figure 10: Flow Chart for Identifying Nodes Beyond All The Branches [4]